TEXAS A&M UNIVERSITY

CSCE 421: Machine Learning

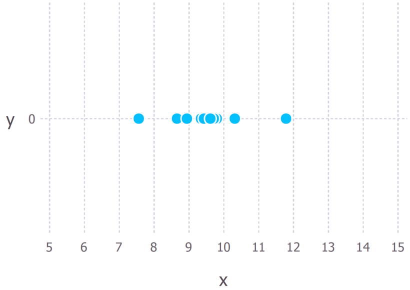
Lecture 6

Overview

- Brief probability review
- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

General Probability Review

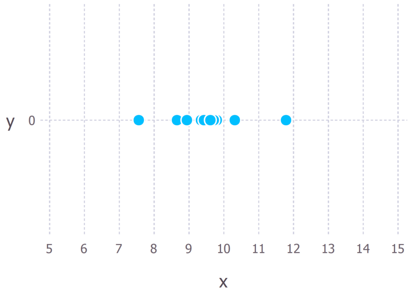
Example: Duration (sec) to answer a Multiple Choice Question



What do you observe?

General Probability Review

Example: Duration (sec) to answer a MCQ

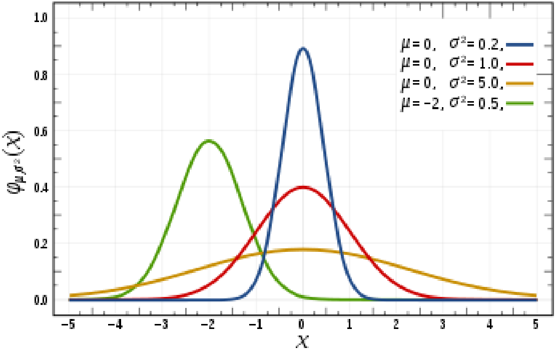


What do you observe?

It is possible that the data are generated from a Gaussian distribution, since most of the points lie in the middle, while some points are scattered to the left and the right.

General Probability Review

Normal distribution

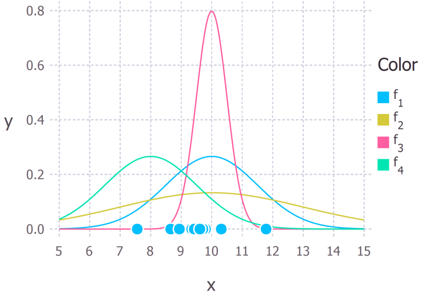


$$x \sim \mathcal{N}(0, \sigma^2) \quad \rightarrow \quad p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Mean μ, variance σ^2, precision $\tau = 1/\sigma^2$

General Probability Review

Which model best describes the data?

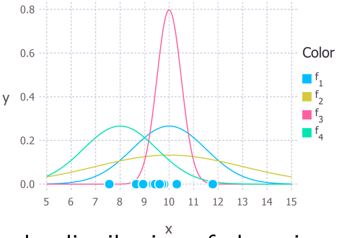


$f_1 \sim \mathcal{N}(10, 2.25),\ f_2 \sim \mathcal{N}(10, 9),\ f_3 \sim \mathcal{N}(10, 0.25),\ f_4 \sim \mathcal{N}(8, 2.25)$

Is there a systematic way to find the distribution that describes "best" the data?

General Probability Review

Which model best describes the data?



- We can calculate the distribution of observing each of the data x_n
  $p(x_n|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x_n-\mu)^2}{2\sigma^2}\right),\ n = 1,\dots,N$
- Find the joint distribution of all data $\mathcal{X}=\{x_1,\dots,x_N\}$ (likelihood)

$$p(\mathcal{X}|\mu,\sigma^2) = p(\{x_1,\dots,x_N\}|\mu,\sigma^2) = \prod_{n=1}^{N} p(x_n|\mu,\sigma^2)$$

$$= \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x_n-\mu)^2}{2\sigma^2}\right)$$

- Find the parameters μ and σ that maximize this joint distribution

Maximum likelihood estimation

- Independent identically distributed sample $\mathcal{X} = \{x_1, \ldots, x_N\}$
- Assume all samples are drawn from the same distribution $p(x|\boldsymbol{\theta})$
- We want to find $\boldsymbol{\theta}$ that makes sampling from $p(x|\boldsymbol{\theta})$ as likely as possible $\rightarrow$ maximize likelihood

$$l(\boldsymbol{\theta}|\mathcal{X}) \equiv p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(x_n|\boldsymbol{\theta})$$

- **Maximum Likelihood estimator (MLE):** the parameter $\boldsymbol{\theta}^{\boldsymbol{MLE}}$ that maximizes the likelihood

$$\boldsymbol{\theta}^{\boldsymbol{MLE}} = \max_{\boldsymbol{\theta}} l(\boldsymbol{\theta}|\mathcal{X})$$

- For the sake of convenience, we take the log-likelihood

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{X}) \equiv \log p(\mathcal{X}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \log p(x_n|\boldsymbol{\theta})$$

Maximum likelihood estimation: Examples

Normal: models a sample from a population with continuous values

- X: Gaussian normal distributed with mean μ and variance σ^2
- **PDF:** $p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
- **MLE estimation:** Sample $\mathcal{X} = \{x_1, \ldots, x_N\}$

$$m = \mu^{MLE} = \frac{\sum_{n=1}^{N} x_n}{N} \quad s^2 = (\sigma^2)^{MLE} = \frac{\sum_{n=1}^{N} (x_n - \mu^{MLE})^2}{N}$$

i.e. the MLE estimate for the *population mean* is the *sample mean*

Note: Not all continuous variables follow the normal distribution, we might have to perform statistical tests for that

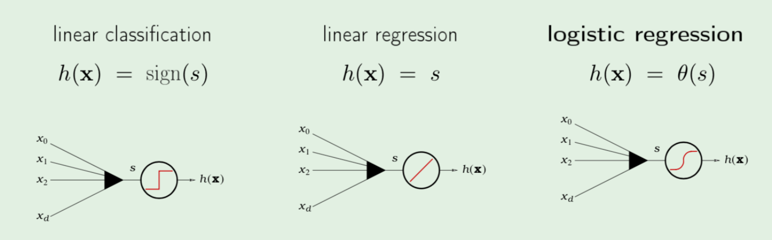
Overview

- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Why the sigmoid function?

Three linear models that we have seen so far

$$s = \mathbf{w}^T\mathbf{x} = \sum_{d=1}^{D} w_d x_d$$

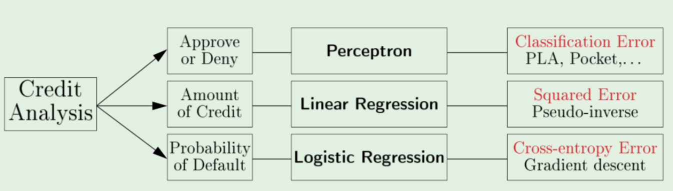


With logistic regression, we can find a soft threshold and model uncertainty.

Why the sigmoid function?

Three linear models that we have seen so far

Example of credit analysis



Bernoulli distribution

The pdf of a single experiment asking a yes/no question

- $Y \sim \text{Bernoulli}(\theta)$, where $Y \in \{0, 1\}$
- $p(y|\theta) = \theta^{\mathbb{I}(y=1)}(1-\theta)^{\mathbb{I}(y=0)} = \begin{cases} \theta & y = 1 \\ 1-\theta & y = 0 \end{cases}$
- e.g. coin toss experiment

Logistic Regression

Parametric classification method (not regression)

Sometimes referred as "generalization" of linear regression because

- We still compute a linear combination of feature inputs, i.e. $\mathbf{w}^T\mathbf{x}$
- Instead of predicting a continuous output variable from $\mathbf{w}^T\mathbf{x}$
  - We pass $\mathbf{w}^T\mathbf{x}$ through a function $\mu(\mathbf{w}^T\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})$

$$\mu(\eta) = \sigma(\eta) = \frac{1}{1 + e^{-\eta}}, \ \ 0 \leq \mu(\eta) \leq 1$$

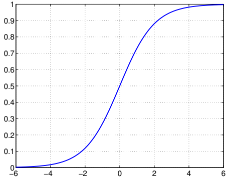
- The above can be considered as the parameter θ of a Bernoulli distribution

$$p(y|\mathbf{x}, \mathbf{w}) = \text{Ber}(y|\mu(\mathbf{w}^T\mathbf{x}))$$

The output belongs to class 1 ($y = 1$) with probability $\theta = \mu(\mathbf{w}^T\mathbf{x})$ and to class 0 ($y = 0$) with probability $1 - \theta = 1 - \mu(\mathbf{w}^T\mathbf{x})$.

Why the sigmoid function?

$$\sigma(\eta) = \frac{1}{1+e^{-\eta}} = \frac{e^{\eta}}{1+e^{\eta}}$$



Very nice properties

- Bounded between 0 and 1 $\leftarrow$ thus interpretable as a probability
- Monotonically increasing $\leftarrow$ thus can be used for classification rules
  - $\sigma(\eta) > 0.5$, positive class (y=1)
  - $\sigma(\eta) \leq 0.5$, positive class (y=0)
- Nice computational properties for optimizing criterion function

Logistic Regression: Representation

Setup for two classes

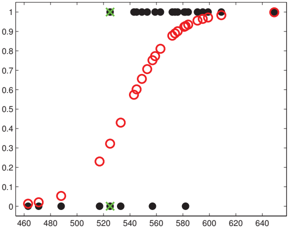
- Input: $\mathbf{x} \in \mathbb{R}^D$
- Output: $y \in \{0, 1\}$
- Training data: $\mathcal{D}^{train} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_N}, y_N)\}$
- Model:

$$p(y = 1|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x}), \quad \sigma(\eta) = \frac{1}{1 + e^{-\eta}}$$

$$y = \begin{cases} 1, & p(y = 1|\mathbf{x}, \mathbf{w}) > 0.5 \\ 0, & \text{otherwise} \end{cases}$$

- Model parameters: weights $\mathbf{w}$

Logistic Regression



Classification task: whether a student passes or not the class

Features: SAT scores

Data: SAT scores v.s. fail/pass (y=0/1) (solid black dots)

Logistic regression:

- Assigns each score to "pass" probability (open red circles)
- If $p(y = 1|x) > 0.5$, then decides $y(x) = 1$. Otherwise, $y(x) = 0$.

Logistic Regression: Evaluation

Data likelihood for 1 training sample

$$p(y_n|\mathbf{x_n}, \mathbf{w}) = \left\{ \begin{array}{ll} \sigma(\mathbf{w}^T\mathbf{x_n}), & y_n = 1 \\ 1 - \sigma(\mathbf{w}^T\mathbf{x_n}), & y_n = 0 \end{array} \right\} = \left[\sigma(\mathbf{w}^T\mathbf{x_n})\right]^{y_n} \left[1 - \sigma(\mathbf{w}^T\mathbf{x_n})\right]^{1-y_n}$$

Data likelihood for all training data

$$L(\mathcal{D}|\mathbf{w}) = \prod_{n=1}^{N} p(y_n|\mathbf{x_n}, \mathbf{w}) = \prod_{n=1}^{N} \left[\sigma(\mathbf{w}^T\mathbf{x_n})\right]^{y_n} \left[1 - \sigma(\mathbf{w}^T\mathbf{x_n})\right]^{1-y_n}$$

Cross-entropy error (negative log-likelihood)

$$\begin{aligned} \mathcal{E}(\mathbf{w}) &= -\log L(\mathcal{D}|\mathbf{w}) \\ &= -\sum_{n=1}^{N} \left\{ y_n \log\left[\sigma(\mathbf{w}^T\mathbf{x_n})\right] + (1 - y_n) \log\left[1 - \sigma(\mathbf{w}^T\mathbf{x_n})\right] \right\} \end{aligned}$$

Logistic Regression: Optimization

Cross-entropy error (negative log-likelihood)

$$\mathcal{E}(\mathbf{w}) = -\sum_{n=1}^{N} \left\{ y_n \log\left[\sigma(\mathbf{w}^T\mathbf{x_n})\right] + (1-y_n)\log\left[1-\sigma(\mathbf{w}^T\mathbf{x_n})\right] \right\}$$

How to find the weights $\mathbf{w}$ of the logistic regression?
We can maximize data likelihood or minimize cross-entropy error

$$\mathbf{w}^* = \min_{\mathbf{w}} \mathcal{E}(\mathbf{w})$$

No closed-form solution $\rightarrow$ approximate methods, e.g. Gradient Descent.

$$\mathbf{w} := \mathbf{w} - \alpha(k)\cdot\nabla\mathcal{E}(\mathbf{w}), \quad \frac{\vartheta\mathcal{E}(\mathbf{w})}{\vartheta w_d} = \sum_{n=1}^{N} \underbrace{\left(\sigma(\mathbf{w}^T\mathbf{x_n}) - y_n\right)}_{\text{error}} x_{nd}$$

$\mathcal{E}(\mathbf{w})$ is convex, i.e. has a global minimum (positive definite Hessian).

Overview

- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?Non-
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Multi-class logistic regression

- Suppose we need to predict multiple classes/outcomes $1, \ldots, C$
  - weather prediction: rainy, cloudy, shiny
  - optical digit/character recongition: 0-9 or 'a'-'z'
- 2-class: probability of $\mathbf{x}$ belonging to class 1
  $p(y=1|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x}), \;\; \sigma(\eta) = \frac{1}{1+e^{-\eta}} = \frac{e^n}{1+e^n}$
- How could we generalize to C classes?
  - One way could be $p(y=c|\mathbf{x}, \mathbf{w_c}) = \sigma(\mathbf{w_c}^T\mathbf{x}) = \frac{e^{\mathbf{w_c}^T\mathbf{x}}}{1+e^{\mathbf{w_c}^T\mathbf{x}}}$
  - This would not work, because each $p(y=c|\mathbf{x}, \mathbf{w_c}) \in [0,1]$ independently
  - And we need $\sum_{c=1}^{C} p(y=c|\mathbf{x}, \mathbf{w_c}) \in [0,1]$
- But we can do the following (softmax function or conditional logit model)
  $p(y=c|\mathbf{x}, \mathbf{w_c}) = \frac{e^{\mathbf{w_c}^T\mathbf{x}}}{\sum_{c=1}^{C} e^{\mathbf{w_c}^T\mathbf{x}}} = \frac{e^{\mathbf{w_c}^T\mathbf{x}}}{e^{\mathbf{w_1}^T\mathbf{x}} + \ldots + e^{\mathbf{w_C}^T\mathbf{x}}}$
  $\sum_{c=1}^{C} p(y=c|\mathbf{x}, \mathbf{w_c}) = 1$

Multi-class logistic regression

- Input: $\mathbf{x} \in \mathbb{R}^D$
- Output: $y \in \{1, 2, \ldots, C\}$
- Training data: $\mathcal{D}^{train} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_N}, y_N)\}$
- Model:

$$p(y = c|\mathbf{x}, \mathbf{w_c}) = \frac{e^{\mathbf{w_c}^T \mathbf{x}}}{\sum_{c=1}^{C} e^{\mathbf{w_c}^T \mathbf{x}}}$$

$$y = arg \max_{c=1,\ldots,C} p(y = c|\mathbf{x}, \mathbf{w_c})$$

- Model parameters: weights $\mathbf{w_1}, \ldots, \mathbf{w_C}$

Multi-class logistic regression

Binary logistic regression is a special case of multi-class

From $p(y=c|\mathbf{x},\mathbf{w_c}) = \frac{e^{\mathbf{w_c}^T\mathbf{x}}}{\sum_{c=1}^{C} e^{\mathbf{w_c}^T\mathbf{x}}}$ for $c=\{0,1\}$, we get

$$p(y=1|\mathbf{x},\mathbf{w_c}) = \frac{e^{\mathbf{w_1}^T\mathbf{x}}}{e^{\mathbf{w_0}^T\mathbf{x}} + e^{\mathbf{w_1}^T\mathbf{x}}} = \frac{1}{e^{\mathbf{w_0}^T\mathbf{x}-\mathbf{w_1}^T\mathbf{x}}+1} = \frac{1}{1+e^{(\mathbf{w_0}-\mathbf{w_1})^T\mathbf{x}}}$$

Same as $p(y=1|\mathbf{x},\mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x})$ with $\mathbf{w} = \mathbf{w_0} - \mathbf{w_1}$

Multi-class logistic regression: Optimization

Discriminative Approach

- We will change $y_n \in \mathbb{R}$ to a C-dimensional vector

$$\mathbf{y_n} = [y_{n1}, \ldots, y_{nC}]^T \in \mathbb{R}^C$$

$$y_{nc} = \begin{cases} 1, & \text{if } \ y_n = c \\ 0, & \text{otherwise} \end{cases}$$

e.g. if $y_n = 3$ then $\mathbf{y_n} = [0, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^C$

- We will maximize the likelihood

$$L(\mathcal{D}|\mathbf{w_1}, \ldots, \mathbf{w_C}) = \prod_{n=1}^{N} p(\mathbf{y_n}|\mathbf{x_n})$$

$$= \prod_{n=1}^{N} \left(p(y_{n1} = 1|\mathbf{w_1}, \ldots \mathbf{w_C})^{y_{n1}} \ldots p(y_{nC} = 1|\mathbf{w_1}, \ldots \mathbf{w_C})^{y_{nC}} \right)$$

Multi-class logistic regression: Optimization

Data-likelihood

$$
\begin{aligned}
L(\mathcal{D}|\mathbf{w_1},\ldots,\mathbf{w_C}) &= \prod_{n=1}^{N} p(y_n|\mathbf{x_n}) \\
&= \prod_{n=1}^{N} \left(p(y_{n1}=1|\mathbf{w_1},\ldots\mathbf{w_C})^{y_{n1}}\ldots p(y_{nC}=1|\mathbf{w_1},\ldots\mathbf{w_C})^{y_{nC}}\right) \\
&= \prod_{n=1}^{N}\prod_{c=1}^{C} p(y_{nc}=1|\mathbf{w_1},\ldots\mathbf{w_C})^{y_{nc}}
\end{aligned}
$$

Cross-entropy error

$$
\mathcal{E}(\mathbf{w_1},\ldots,\mathbf{w_C}) = -\sum_{n=1}^{N}\sum_{c=1}^{C} y_{nc}\log p(y_{nc}=1|\mathbf{w_1},\ldots\mathbf{w_C})
$$

Multi-class logistic regression: Optimization

Cross-entropy error

$$\mathcal{E}(\mathbf{w_1}, \ldots, \mathbf{w_C}) = -\sum_{n=1}^{N}\sum_{c=1}^{C} y_{nc} \log p(y_{nc} = 1 | \mathbf{w_1}, \ldots \mathbf{w_C})$$

- Optimization with gradient descent, convex function
- Computational details are out of scope
- But the gradient vector w.r.t. each weight $\mathbf{w_c}$ looks like this

$$\nabla \mathcal{E}_{\mathbf{w_c}} = \sum_{n=1}^{N} \underbrace{[p(y_{nc} = 1 | \mathbf{w_1}, \ldots \mathbf{w_C}) - y_{nc}]}_{\text{error for class c}} \mathbf{x_n}$$

- Similar to binary logistic regression $\rightarrow$ General property of exponential family distributions

Overview

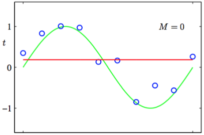
- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Overfitting

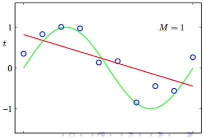
Example: Non-linear regression $y = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$

Samples from a sine function $x_i = \sin(t_i)$, $t_i \sim \text{Uniform}(0, 2\pi)$

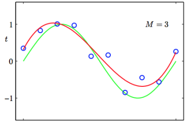
$M = 0$ underfitting



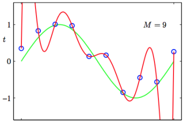
$M = 1$ underfitting



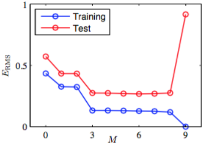
$M = 3$



$M = 9$ overfitting



	$M = 0$	$M = 1$	$M = 3$	$M = 9$
w_0	0.19	0.82	0.31	0.35
w_1		-1.27	7.99	232.37
w_2			-25.43	-5321.83
w_3			17.37	48568.31
w_4				-231639.30
w_5				640042.26
w_6				-1061800.52
w_7				1042400.18
w_8				-557682.99
w_9				125201.43

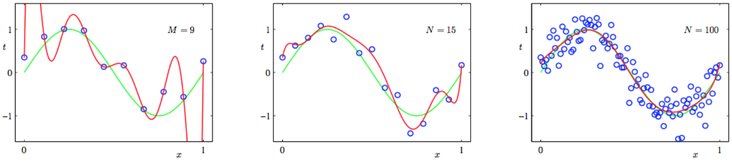


As model becomes more complex, performance on training keeps improving while on test data improve first and deteriorate later.

The larger a coefficient w_i, the easier for the model to "swing" in that dimension, increasing chance to fit more noise.

How can we avoid overfitting?

One solution: Use more training data



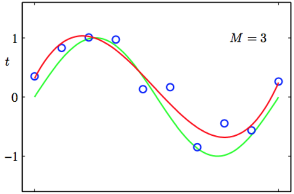
What if we don't have a lot of data?

Another solution: Use less features (e.g. feature selection algorithms)
Intuitively, this will reduce the complexity of the model, therefore it is likely to result in less overfitting.

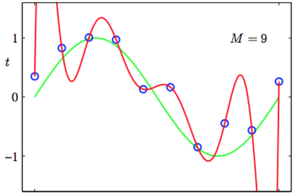
How can we avoid overfitting?

A more general solution: Regularization

$M = 3$



$M = 9$



$$y = w_0 + w_1 x + w_2 x^2 \qquad y = w_0 + w_1 x + w_2 x^2 + \ldots + w_9 x^9$$

How about penalizing and making small $w_3, \ldots, w_9$?

The cost function to be minimized would become:

$J(\mathbf{w}) = RSS(\mathbf{w}) + w_3^2 + \ldots w_9^2$

But we may not know in advance which parameters we want to penalize

$\rightarrow$ So we can penalize them all

How can we avoid overfitting?

A more general solution: Regularization

Suppose we have a learning model whose evaluation criterion $EC(\mathbf{w})$ we want to optimize with respect to weights $\mathbf{w} = [w_1, \ldots, w_D]^T$

- $J(\mathbf{w}) = EC(\mathbf{w}) + \lambda \sum_{d=1}^{D} w_d^2 = EC(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$
  $\rightarrow$ l2-norm regularization
- $J(\mathbf{w}) = EC(\mathbf{w}) + \frac{\lambda}{N} \sum_{d=1}^{D} w_d^2$
  (as #data N increases, we need to worry less about overfitting)
- $J(\mathbf{w}) = EC(\mathbf{w}) + \lambda \sum_{d=1}^{D} \|w_d\| = EC(\mathbf{w}) + \lambda \|\mathbf{w}\|$
  $\rightarrow$ l1-norm regularization

Evaluation criterion $EC(\mathbf{w})$ can be RSS or log-likelihood for linear regression, negative cross-entropy for logistic regression, etc.

$\lambda \geq 0$ is the model complexity penalty

Overview

- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Regularization for Non-Linear Regression

l2-norm regularization

Linear: $J(\mathbf{w}) = RSS(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2$

Non-linear: $J(\mathbf{w}) = RSS(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2 = (\mathbf{y} - \boldsymbol{\Phi}\mathbf{w})^T(\mathbf{y} - \boldsymbol{\Phi}\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2$

Closed-form solution:

Linear: $\mathbf{w}^* = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{D\times D})^{-1}\mathbf{X}^T\mathbf{y}$

Non-linear: $\mathbf{w}^* = (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\mathbf{I}_{D\times D})^{-1}\boldsymbol{\Phi}^T\mathbf{y}$

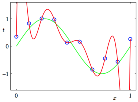
The above reduces to ordinary least squares (OLS) solution when $\lambda = 0$ (see handout for derivation)

Regularization for Non-Linear Regression

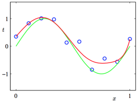
Question: Assume a set of samples generated from a sine function $x_i = \sin(t_i)$ (green line), modeled with **regularized** non-linear regression $y = w_0 + w_1 x + \ldots + w_9 x^9$. How does the resulting model (red line) look as we increase the amount of regularization λ?

A)

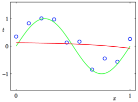
$\lambda = 0$



$\lambda = e^{-10}$

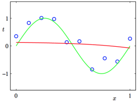


$\lambda = 1$

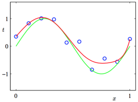


B)

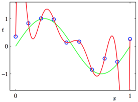
$\lambda = 0$



$\lambda = e^{-10}$

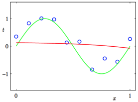


$\lambda = 1$

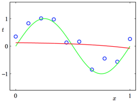


C)

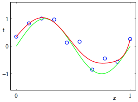
$\lambda = 0$



$\lambda = e^{-10}$



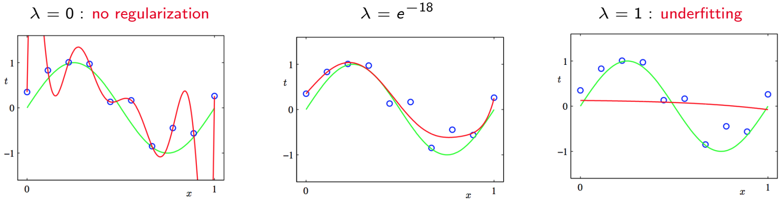
$\lambda = 1$



Regularization for Non-Linear Regression

Question: Assume a set of samples generated from a sine function $x_i = \sin(t_i)$ (green line), modeled with **regularized** non-linear regression $y = w_0 + w_1 x + \ldots + w_9 x^9$. How does the resulting model (red line) look as we increase the amount of regularization λ?

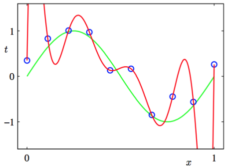
The correct answer is A



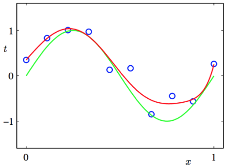
Overfitting is reduced with the help of increasing regularizers

Regularization for Non-Linear Regression

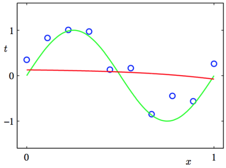
$\lambda = 0$: no regularization

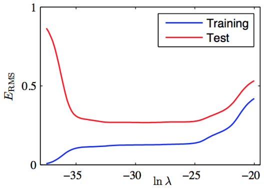


$\lambda = e^{-18}$



$lambda = 1$





	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
w_0	0.35	0.35	0.13
w_1	232.37	4.74	-0.05
w_2	-5321.83	-0.77	-0.06
w_3	48568.31	-31.97	-0.06
w_4	-231639.30	-3.89	-0.03
w_5	640042.26	55.28	-0.02
w_6	-1061800.52	41.32	-0.01
w_7	1042400.18	-45.95	-0.00
w_8	-557682.99	-91.53	0.00
w_9	125201.43	72.68	0.01

For a complex model ($M = 9$), training error increases with increasing regularization.

Overview

- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Regularization for Logistic Regression

l2-norm regularization

$$\mathcal{E}(\mathbf{w}) = -\sum_{n=1}^{N} \left\{ y_n \log\left[\sigma(\mathbf{w}^T \mathbf{x_n})\right] + (1 - y_n) \log\left[1 - \sigma(\mathbf{w}^T \mathbf{x_n})\right] \right\} + \lambda \|\mathbf{w}\|_2^2$$

$$\nabla \mathcal{E}(\mathbf{w}) = \sum_{n=1}^{N} \left(\sigma(\mathbf{w}^T \mathbf{x_n}) - y_n \right) \mathbf{x_n} + 2\lambda \mathbf{w}$$

$$\mathbf{H} = \sum_{n=1}^{N} \underbrace{\sigma(\mathbf{w}^T \mathbf{x_n})}_{\in[0,1]} \cdot \underbrace{\left(1 - \sigma(\mathbf{w}^T \mathbf{x_n})\right)}_{\in[0,1]} \cdot \underbrace{\left(\mathbf{x_n} \cdot \mathbf{x_n}^T\right)}_{\in \mathcal{R}^{D \times D}} + \lambda \mathbf{I}_{D \times D}$$

(see handout for derivations)

Overview

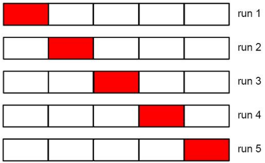
- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

Overview

- Logistic Regression
  - Representation and Intuition
  - Evaluation through maximum-likelihood
  - Optimization through gradient descent
  - Convexity of evaluation criterion
- Multiclass logistic regression
  - Representation (derivation based on 2-class)
  - Evaluation through cross-entropy error
- Regularization
  - Why do we need it?
  - Non-Linear Regression
  - Logistic Regression
  - How to choose the right amount of regularization

How to choose the right amount of regularization?

- We cannot tune λ on the train set. Why?
- λ is a hyper-parameter and we can tune it by:
  - keeping out a hold-out-set independent of train and test sets
  - doing cross-validation
  - similar procedure to choosing K for K-NN



Recipe for cross-validation for choosing λ

- Split train data into S equal parts, each noted as $\mathcal{D}_s^{train}$, $s = 1,...,S$
- For each hyperparameter value (e.g. $\lambda = 10^{-5}, 10^{-4}, \dots$)
    - For each $s = 1, \dots, S$
        - Train model using $\mathcal{D}^{train} \setminus \mathcal{D}_S^{train}$
        - Evaluate model performance (noted as E_s) on $\mathcal{D}_s^{train}$
    - Compute average performance for current hyperparameter $E = \frac{1}{s} \sum_{s=1}^{S} E_s$
- Chose the hyperparameter corresponding to best average performance E
- Use the best hyperparameter to train on a model using all $\mathcal{D}^{train}$
- Evaluate the last model on $\mathcal{D}^{test}$

What have we learnt so far

Logistic Regression

- Linear combination of input features $\mathbf{w}^T\mathbf{x}$
- Transform through sigmoid function $\sigma(\mathbf{w}^T\mathbf{x}) \rightarrow$ interpretable as probability
- Decision rule based on whether $\sigma(\mathbf{w}^T\mathbf{x}) \lesseqgtr 0.5$
- Evaluation through data likelihood, or cross-entropy error

$$\mathcal{E}(\mathbf{w}) = -\sum_{n=1}^{N} \left\{ y_n \log \left[\sigma(\mathbf{w}^T\mathbf{x_n}) \right] + (1 - y_n) \log \left[1 - \sigma(\mathbf{w}^T\mathbf{x_n}) \right] \right\}$$

- Optimization through gradient descent

What have we learnt so far

Multinomial Regression

- Conditional logit model: $p(y = c|\mathbf{x}, \mathbf{w_c}) = \frac{e^{\mathbf{w_c}^T \mathbf{x}}}{\sum_{c=1}^{C} e^{\mathbf{w_c}^T \mathbf{x}}}$
- Similar to 2-class logistic regression
  - compute negative cross-entropy and perform gradient descent

Regularization

- Method to avoid overfitting
- Penalize large weights with l1 or l2-norm regularization
  $J(\mathbf{w}) = EC(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$

Readings: Alpaydin 10.7; Abu-Mostafa 3